第 41 卷
2005 年 10 月

南京大学学报(自然科学)
JOURNAL OF NANJING UNIVERSITY
(NATURAL SCIENCES)

Vol. 41
Oct., 2005

控制软件质量预测中两类错误比率的比较研究*

邢 飞[1,2], 郭 平[1], 吕荣聪[3]

(1. 北京师范大学计算机科学系,北京,100875;
2. 中国科学院计算机科学重点实验室,北京,100080;
3. 香港沙田香港中文大学计算机科学与工程学系,香港)

摘 要: 软件质量预测是通过建立软件内部属性(复杂性度量)与外部属性(缺陷数)之间客观定量的联系,来帮助软件开发者和管理者较早地检测出出错率高的模块,避免将错误带入软件生命周期后期.在软件质量预测中常会出现两类错误,在实际应用中,第二类错误所引发的后果往往要比第一类错误要严重得多,因此降低第二类错误率是非常必要的.探讨了 3 种控制软件质量预测中两类错误比率的方法:带风险特性的支持向量机,偏置支持向量机和基于最小风险的贝叶斯决策.实验结果表明,限定第二类错误的情况下,带风险特性的支持向量机在总体的预测性能上是最优的,但其对第二类错误的调节幅度是有限的.基于最小风险的贝叶斯决策与其相反,它可以将第二类错误控制到很小,但它的总体分类性能也是三种方法中最差的.而偏置支持向量机的性能介于上述两者之间.由此可以根据不同的应用需求来选用一种适合的方法来控制两类错误的比率.

关键词: 复杂性度量,软件质量预测,支持向量机,贝叶斯决策,两类错误
中图分类号: TP 301

Comparison of Methods of Controlling Two Types of Error Ratio Applied to Software Reliability Prediction

Xing Fei[1,2], *Guo Ping*[1,2], *Lyu R. Michael*[3]

(1. Department of Computer Science, Beijing Normal University, Beijing, 100875, China;
2. Laboratory for Computer Science, The Chinese Academy of Sciences, Beijing, 100080, China;
3. Department of Computer Science & Engineering, The Chinese University of Hong Kong, Hong Kong, China)

Abstract: Software quality prediction is to help software developers and managers indicate the fault - prone modules early and avoid leaving faults to later software life cycle by building the relationship between some complexity metrics and the number of faults. In classifying a module to be fault - prone or non fault - prone, there are two types of errors that can be made in the partition. In the real project, Type II error has more serious implications, so it is necessary to reduce Type II error. In this paper, three methods of adjusting the weights between two types of error, namely support vector machine (SVM) with risk feature, Biased SVM and Bayesian decision with the minimum risk are applied to software quality prediction. The Experimental results show that the performance of classification using SVM with risk feature is best when restricting the Type II error to be a constant. But it cannot reduce the Type II error to a relative low value. The case using Biased SVM is even opposite to SVM with risk feature that it can reduce the Type II error to a very low value even to zero. But its performance of classification is the lowest. As for Biased SVM, it achieves moderate performance.

* 基金项目:国家自然科学基金(60275002),国家 863 高科技项目(2003AA133060)

Consequently we can choose an appropriate method to build models according to different conditions in order to control the ratio of two types of error.

Key words: complexity metrics, software quality prediction, support vector machines, Bayesian decision, two types of error

当前计算机软件的应用范围越来越广泛,软件的规模也越来越大.因此,软件质量问题是软件工程领域有待于解决的迫切问题.在过去的研究中人们已经建立了很多实用有效的软件质量预测模型.通过预测模型,可以建立起软件内部属性(复杂性度量)与外部属性(缺陷数)之间客观定量的联系,它可以帮助软件的开发者和管理者在软件生命周期的早期投入更多的注意力在那些出错率较高的模块上,避免将软件生命周期中早期的问题带入后期,从而有效地改善软件质量.

在预测软件质量时,精确预测出软件的每个模块的缺陷数往往是不必要的,所以常把软件模块分成两大类:易出错模块(fault-prone)和不易出错模块(non fault-prone).目前已经有很多技术应用于软件模块分类,其中支持向量机(Support Vector Machine, SVM)技术作为一种出色的机器学习方法,已经成功地应用于软件质量预测.通过质量预测模型的建立,易出错的软件模块可以在软件生命周期中更早阶段被审核确认,避免了将错误带入软件开发的后期,从而有效地降低了软件开发成本,提高了软件的可靠性.

在对软件模块分类中可能出现两类错误,第一类错误是将原本不易出错的模块分到易出错模块的类中,第二类错误是将易出错的模块分到不易出错模块的类中.对于这两类错误,第一类错误只会导致对一些模块进行额外的测试,而第二类错误则可能在软件开发后期导致一些无法预料的错误.因为一个软件产品表面上看起来比它实际要好,所以测试往往不能专注于那些最需要测试的模块.目前大多数质量预测模型对于两类模块的处理是平等的.例如,标准支持向量机是寻找一个最优的分类超平面使其可以将两类模块正确地分开并且保证两类的分类空隙最大.因此它只能保证整体分类正确率达到最优,而忽略了第二类错误的风险.

本文探讨了三种控制软件质量预测中两类错误比率的方法:带风险特性的支持向量机,偏置支持向量机和基于最小风险的贝叶斯决策.其中前两者是基于支持向量机理论的,而后者是目前最常用的一种基于风险的分类方法.通过对这三种分类模型的分析,可以达到控制第二类错误的目的,避免了将一些严重的错误带入软件开发后期.

1　支持向量机

支持向量机是 Vapnik 等人提出的一类新型机器学习方法[1],由于其出色的学习性能,该技术已成为机器学习领域的研究热点,并在很多方面都得到了成功的应用[2].

1.1　理论基础　SVM 是以结构化风险最小化(SRM)代替常用的经验风险最小化(ERM)作为优化准则,可以在理论上取得更好的推广性能.为了最小化期望风险的上界,SVM 在固定学习机经验风险的条件下最小化 VC 置信度.SVM 的基本思想是对于非线性可分样本,将其输入向量经非线性变换映射到另一个高维空间 Z 中,在变换后的空间中寻找一个最优的分界面(超平面),使其推广能力最好.下面以两类模式的分类为例说明其基本原理.

设线性可分的样本集有 n 个样本(x_i, y_i),其中 $i=1,2,\cdots,n$, $x\in R^d$. $y\in\{-1,1\}$是类别标号.在高维空间中,将两类样本无错分开的分类超平面满足

$$w\cdot x+b=0, w\in R^N, b\in R \quad (1)$$

通过对向量系数 w 进行归一化,可以使所有样本满足$|g(x_i)|\geqslant 1$,这样分类间隔就等于

$2/\|w\|$，因此使分类间隔最大实际上就是使 $\|w\|$ 最小；而要求分类线对大多数样本正确分类，就要满足

$$y_i(\mathbf{x}_i \cdot \mathbf{w} + b) \geqslant 1 - \xi_i, i = 1,2,\cdots,n \quad (2)$$

根据上述讨论，最优分类面问题可表示成如下的约束优化问题，即在式(2)的约束下，求函数

$$\Phi(w,\xi) = \frac{1}{2}\|w\|^2 + C\left(\sum_{i=1}^{n}\xi_i\right) \quad (3)$$

的最小值. 可利用 Lagrange 函数把原问题转化为较简单的 Wolfe 对偶问题：在约束条件

$$\sum_{i=1}^{n} y_i\alpha_i = 0, 0 \leqslant \alpha_i \leqslant C, i = 1,2,\cdots,n \quad (4)$$

之下对 α_i 求解式下列函数的最大值：

$$Q(\alpha) = \sum_{i=1}^{n}\alpha_i - \frac{1}{2}\sum_{i,j=1}^{n}\alpha_i\alpha_j y_i y_j (x_i \cdot x_j)$$

求解上述问题后得到的最优分类函数是：

$$f(x) = \mathrm{sgn}\{(w \cdot x_i) + b\} = \mathrm{sgn}\left\{\sum_{i=1}^{n}\alpha_i y_i (x_i \cdot x) + b\right\} \quad (6)$$

通过上述讨论可以看出，最优分类识别函数 $f(x)$ 只包含待测样本与支持向量之间的内积. 因此对于非线性可分的特征空间，考虑通过一个非线性映射 T 将特征向量 x 映射到高维线性特征空间 F 中. 高维线性空间中的内积可以定义为：

$$K(x_i, x_j) = T(x_i) \cdot (T(x_j)) \quad (7)$$

支持向量机方法的优点在于没有必要知道映射 T 的具体形式，而只需定义高维空间中的内积运算 $K(x_i, x_j)$ 即可，这样即使变换后空间维数增加很多，计算的复杂度也没有太大的变化.

1.2 带风险特性的支持向量机 在实际的问题中，两类错误的风险往往是不同的. 因此，我们在标准支持向量机基础上引入风险机制，通过调整错误惩罚因子 C 来达到控制风险的目的. 假设训练样本中前 k 个属于第 1 类，后 $l-k+1$ 个属于第 2 类，则公式(3)变为：

$$\Phi(w,\xi) = \frac{1}{2}\|w\|^2 + C_1\left(\sum_{i=1}^{k}\xi_i\right) + C_2\left(\sum_{i=k-1}^{l}\xi_i\right) \quad (8)$$

其中 C_1 和 C_2 分别表示第 1 类和第 2 类的错误惩罚因子.

求解式(8)的最小值所得的结果类似于上述标准的支持向量机，唯一区别是拉格朗日乘数算子边界变为：

$$\sum_{i=1}^{n} yiai = 0, 0 \leqslant \alpha_i \leqslant C_1, i = 1,2,\cdots,k$$
$$0 \leqslant \alpha_i \leqslant C_2, i = k+1, k+2,\cdots,1 \quad (9)$$

1.3 偏置支持向量机 偏置支持向量机(Biased Support Vector Machine, BSVM)是最近提出的一种支持向量机的改进形式[3]，可以处理两类训练样本不平衡的情况. 它的基本理论是由单类支持向量机(One-class SVM)演变得出的. BSVM 的目标是找到两个超球面，要求里面的超球面尽量包含大部分某一类样本，而外面的超球面尽量不包含另一类样本，而且要使两个超球面的距离最大. 这个任务可以转化成如下数学问题：

$$\min_{R\in R,\xi\in R,\rho\in R} bR^2 - \rho + \frac{1}{nv}\sum_{i=1}^{n}\xi_i \quad (10)$$

$$s \cdot t \cdot \ y_i(\|\Phi(x_i) - c\|^2 - R^2) \leqslant -\rho + \xi_i$$
$$b \geqslant 0, \xi_i \geqslant 0, \rho \geqslant 0, 0 \leqslant v \leqslant 1 \quad (11)$$

其中 ξ_i 是松弛因子，$\Phi(x_i)$ 是映射函数，c 和 R 分别是超球面的球心和半径，ρ 是两个球面的间距，b 是控制偏置的参数，v 是调节支持向量个数与球面间距之间权重的参数.

通过引入拉格朗日乘法算子，可以得到如下对偶形式：

$$\max_{\alpha}\sum_{i}\alpha_i y_i k(\mathbf{x}_i, \mathbf{x}_i) - \frac{1}{b}\sum_{i,j}\alpha_i\alpha_j y_i y_j k(x_i, x_j) \quad (12)$$

$$s \cdot t \cdot \ \sum_{i}\alpha_i y_i = b, 0 \leqslant \alpha_i \leqslant \frac{1}{nv}, \sum_{i}\alpha_i \geqslant 1 \quad (13)$$

上式可以通过二次规划技术来解决. 由此看出，可以通过调整参数 b 来控制两类样本的权重.

2　基于最小风险的贝叶斯决策

在贝叶斯决策中,基于最小错误率的贝叶斯决策是保证错误率达到最小的决策,它不考虑决策带来的损失,而基于最小风险的贝叶斯决策是将风险因素考虑到决策中.定义 x 为 d 维随机向量,状态空间由 c 个自然状态 ω_i 组成,决策空间由 a 个决策 α_i,损失函数 $\lambda(\alpha_i, \omega_i)$ 表示当真实为状态 ω_i 而采取的决策为 α_i 时所带来的风险.给定观测值 x 在采取决策 α_i 情况下的条件期望损失为:

$$R(\alpha_i \mid x) = E[\lambda(\alpha_i, \omega_j)] = \sum_{j=1}^{c} \lambda(\alpha_i, \omega_j) P(\omega_i \mid x) \quad i = 1, 2, \cdots, a \tag{14}$$

则基于最小风险的贝叶斯决策规则为:

$$\text{若 } R(\alpha_k \mid x) = \min_{i=1,\cdots,a} R(\alpha_i \mid \mathbf{x}), \text{则 } \alpha = \alpha_k \tag{15}$$

3　实验结果分析比较

实验中,采用的实际数据是医学图像处理系统(MIS)软件,该数据来自参考文献[4]中的光盘.MIS 系统包含了约 4 500 个模块,总计约 400 000 行代码,代码以 Pascal, FORTRAN,汇编和 PL/M 语言编写.在实验中用到的数据是从这些模块中随机抽取的 390 个以 Pascal, FORTRAN 语言编写的模块的复杂性度量.在软件开发的过程中,一个已经完成设计的软件模块经常会因为其低稳定性,低性能等种种原因被修改甚至重新设计,一个软件模块被要求修改的次数(CRs)常被用作软件模块缺陷数的指示器[5].除 CRs 外,每个软件模块还有 11 项软件度量,它们是:代码总行数 TC(Total lines of code),有效代码行数 CL(Number of code lines),代码中字符数 Cr (Number of characters),注释数目 Cm (Number of comments),注释中字符数 CC (Number of comment characters),有效代码字符数 Co (Number of code characters),Halstead 程序长度 N'[6](N' 为代码中操作数个数与算子个数之和),程序长度的 Halstead 估计 Ne ($Ne = \eta_1 \log_2 \eta_1 + \eta_2 \log_2 \eta_2$,其中 η_1 表示不重复的算子数目,η_2 表示不重复的操作数数目),程序长度的 Jensen 估计 JE[7] ($JE = \log_2 \eta_1! + \log_2 \eta_2!$),McCabe 圈复杂度 M[8]($M = e - n + 2$,其中 e 表示 n 节点控制流图边缘数),Belady 带宽 BW[8]($BW = \frac{1}{n}\sum_i i L_i$,其中 L_i 表示 n 节点嵌套控制流图中第 i 层的结点数).

在建模实验中,定义含有 0 或 1 个 CRs 的模块为不易出错的模块,含有 10～98 个 CRs 的模块为易出错的模块.对于实验中用到的 MIS 数据,不易出错的模块共 114 个,易出错的模块共 89 个.实验中采用 bootstrap[9]技术,每类中随机选取二分之一的样本作为训练样本,剩余的作为检验样本用于计算分类正确率(Correct classification rate, CCR).每个实验都重复 25 次,最后计算出的是 CCR 的均值和方差.

首先采用 1.1 节所描述的带风险特性的支持向量机对软件质量进行预测.实验中,设定参数 C_2 为固定值 20 000,通过调节参数 C_1 来控制第二类错误.表 1 和图 1 显示了实验结果,可以看到随着 C_1 的不断减小,当前的分类面逐渐偏离最优分类面,因此总的分类正确率不断的下降.但此时由于第一类支持向量的权重系数 α_i 的取值范围逐渐变小,因此第二类错误也不断的下降.当 C_1 降至 5000 时,总的分类正确率降为 86.53%,而第二类错误也降至 5.10%.

表 1　带风险特性的 SVM 预测结果

C_1	C_2	CCR	Std	TI error	TII error
20 000	20 000	89.07%	0.020 9	2.06%	8.87%
15 000	20 000	89.07%	0.020 9	2.06%	8.87%
10 000	20 000	89.00%	0.018 9	2.33%	8.67%
8 000	20 000	86.53%	0.027 5	4.90%	7.52%
5 000	20 000	86.53%	0.039 3	11.43%	5.10%

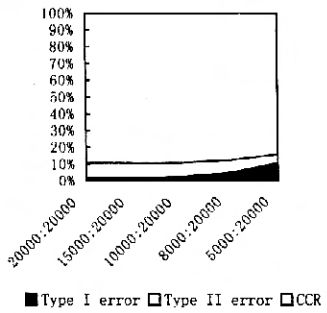

图1 带风险特性的SVM实验结果的图形表示

表2 偏置SVM预测结果

Bias①	CCR	Std	TI error	TII error
20 000	88.22%	0.029 0	3.59%	8.20%
22 000	86.47%	0.043 3	6.88%	6.65%
23 000	84.12%	0.060 4	9.69%	6.20%
25 000	82.29%	0.079 8	12.12%	5.59%
28 000	79.47%	0.107 8	15.67%	4.86%

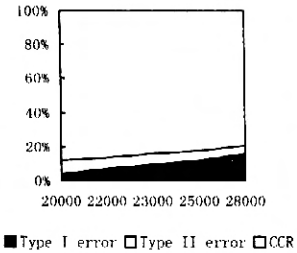

图2 偏置SVM实验结果的图形表示

第二个实验中,采用1.2节所描述的偏置支持向量机对软件质量度量数据进行建模预测.实验中取参数C为固定值20 000,通过调整偏置参数b来控制两类样本支持向量的权重从而达到减小出现第二类错误的概率.从表2和图2中可以看出,随着偏置b的不断增大,当前的分类面逐渐向有利于第二类正确分类的趋势偏移,因此第二类错误不断的下降.但是由于分类面偏离了最优分类面,因此总的分类正确率不断的下降.与带风险特性的支持向量机相比,在降到相同的第二类错误率时,偏置支持向量机的总体分类性能弱于带风险特性的支持向量机.

基于最小风险的贝叶斯决策理论是一种最常用的风险决策方法,作为比较,采用正态分布模型时的最小风险贝叶斯决策,表3和图3中显示了基于不同风险比的最小风险贝叶斯决策分类效果.可以看出,随着第二类错误的风险增大,出现第二类错误的比率逐渐减小至可以忽略不计,但整体的分类正确率也在不断降低,当风险比为1.4:1时,CCR仅为59.98%,由此可以看出降低第二类错误发生概率是以牺牲整体分类正确率和增加第一类错误为代价的.另外,实验结果显示了基于最小风险的贝叶斯决策的特点是可以将第二类错误降至很低,而这是前两种方法错做不到的.但是基于最小风险的贝叶斯决策的总体分类性能是最差的.因此获得较低的第二类错误率是以牺牲总体分类性能为代价的.

表3 基于最小风险的贝叶斯决策实验结果

Risk ratio	CCR	Std	TI error	TII error
1:1	85.94%	0.038 7	7.59%	6.47%
1.1:1	83.80%	0.032 6	11.06%	5.14%
1.2:1	78.73%	0.032 1	17.90%	3.37%
1.3:1	71.31%	0.043 6	27.12%	1.57%
1.4:1	59.98%	0.051 6	39.75%	0.27%

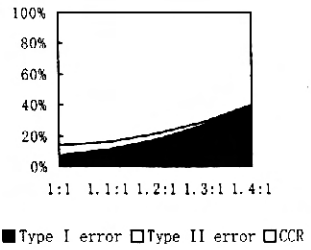

图3 偏置SVM实验结果的图形表示

4 总　结

本文比较了三种控制软件质量预测中两类错误比率的方法:带风险特性的支持向量机,偏置支持向量机和基于最小风险的贝叶斯决策.从实验结果中可以看出,利用这三种技术进行建模,可以达到控制第二类错误的目的,但它们各有自己的特点.在限定第二类错误的情况下,带风险特性的支持向量机在总体的预测性能上是最优的,但其对第二类错误的调节幅度是有限的.基于最小风险的贝叶斯决策与其相反,它可以将第二类错误调节到很小,但它的总体分类性能也是三种方法中最差的.而偏置支持向量机的性能介于两者之间.因此可以根据不同的应用需求来选定一种适合的方法减小风险,相信基于控制第二类错误的软件质量预测模型在实际应用中可以有效地改善软件质量,提高软件的可靠性.

References

[1] Cortes C, Vapnik V. Support Vector Networks. Machine Learning, 1995, 20:273~297.

[2] Xing F, Guo P. Support Vector Regression for Software Reliability Growth Modeling and Prediction. Lecture Notes in Computer Science. Springer-Verlag, 2005, 3 496:925~930.

[3] Hoi C H, Chan C H, Huang K, *et al*. Biased support vector machine for relevance feedback in image retrieval. Proceedings of IJCNN' 2004, IEEE Computer Society Press, 2004, 3 189~3 294.

[4] Lyu M R. Handbook of software Reliability Engineering. McGraw Hill: IEEE Computer Society Press, 1996.

[5] Basili V R, Hutchens D H. An empirical study of a syntactic complexity family. IEEE Transactions on Software Engineering, 1983, SE - 9(6): 664 ~ 672.

[6] Halstead M. Elements of Software Science. North-Holland: New York Elsevier, 1977.

[7] Jensen H, Vairavan K. An experimental study of software metrics for real-time software. IEEE Transactions on Software Engineering, 1994, SE-11(2):231~234.

[8] McCabe T J. A complexity measure. IEEE Transactions on Software Engineering, 1976, SE - 2 (4):308~320.

[9] Efron B, Tibshirani R. An Introduction to the Bootstrap. Chaoman & Hall, 1993,436.

[10] Xing F, Guo P. Early Software Quality Prediction using SVM. Proceeding of 2004 National Software and Applications Conference (NASAC2004). Beijing: China Machine Press, 2004, 207 ~ 211. (邢　飞, 郭　平.基于支持向量机的早期软件质量预测研究. 全国软件与应用学术年会. 北京:机械工业出版社, 2004, 207~211).